SECURITY AND COMMUNICATION NETWORKS
Security Comm. Networks (2016)
Published online in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/sec.1459

RESEARCH ARTICLE

An energy efficient encryption method for secure dynamic WSN

Mohamed Elhoseny[2], Xiaohui Yuan[1]*, Hamdy K. El-Minir[3] and Alaa Mohamed Riad[1]

[1] Department of Computer Science and Engineering, the University of North Texas, Denton, TX, U.S.A.
[2] Department of Information Systems, Mansoura University, Mansoura, Egypt
[3] Department of Electrical Engineering, Sheikh University, Kafr El-Sheikh, Egypt

ABSTRACT

Clustering methods have been developed to improve network life of wireless sensor network (WSN), yet the dynamic nature of sensor clusters and limited memory and processing power make security a much more challenging problem, and most conventional cryptography methods are ill suited to WSNs. In this paper, we propose a novel encryption method to secure data transmission in WSN with dynamic sensor clusters. Our method leverages elliptic curve cryptography algorithm to generate binary strings for each sensor and combines with node ID, distance to the cluster head, and the index of transmission round to form unique 176-bit encryption keys. Using exclusive OR, substitution, and permutation operations, encryption and decryption are achieved efficiently. Compared with the state-of-the-art methods, our simulation results demonstrated that the proposed method exhibited much improved network lifetime and reduced the energy consumption most evenly among all sensor nodes. More importantly, it overcame many security attacks including brute-force attack, HELLO flood attack, selective forwarding attack, and compromised cluster head attack. Copyright © 2016 John Wiley & Sons, Ltd.

KEYWORDS
wireless sensor networks; encryption; energy consumption; secure WSN; elliptic curve cryptography

***Correspondence**
Xiaohui Yuan, College of Information Engineering, China University of Geosciences, Wuhan, China; Department of Computer Science and Engineering, the University of North Texas, Denton, TX, U.S.A.
E-mail: xiaohui.yuan@unt.edu

1. INTRODUCTION

As sensor becomes ubiquitous, secure and efficient wireless sensor network (WSN) is in high demand because sensors are usually deployed in an unattended, harsh field where eavesdropping and network tempering is highly possible. On the other hand, clustering methods have been developed to improve network life of WSN, which has demonstrated great success in the past [1]. The dynamic nature of sensor clusters, however, makes security a much more challenging problem as the communication channels are frequently reorganized. Conventional data encryption and decryption methods to ensure secure transmission demand high computational resources, which could significantly degrade the sensor life. In addition, limited amount of memory and processing power make most conventional cryptography methods ill suited for WSNs.

To ensure data security, many methods have been developed to encrypt data in WSN. Mykletun *et al.* proposed Elliptic Curve Okamoto-Uchiyama (EC-OU) method that uses a public key-based encryption scheme to achieve data concealment [2] and hence consumes less energy. The ciphertext size is rather large as the result of EC-OU method, however, which demands greater amount of memory space in the resource limited WSN applications. Liu *et al.* [3] proposed a block cipher that uses a chaotic S-box algorithm and Substitution-permutation (SP) networks to eliminate the need of floating-point operations and multiplications and allows variable length encryptions. The method achieved low CPU demand for a fairly short key of 32 bits. Light encryption device (LED) [4] was devised based on the lightweight block cipher scheme. To generate a new key in LED, the key is altered with exclusive OR operation at every four round. Yet, LED requires more CPU cycles compared with the conventional cryptographic schemes [5]. Suzaki *et al.* proposed a block cipher method (TWINE) [6] that uses an 80-bit or 128-bit key. The method provides greater security; the weakness of this method is that it is vulnerable to two biclique attacks [7]. Girao *et al.* proposed TinyPEDS method [8] for secure data aggregation at the cluster heads using homomorphic encryption. Notes are pre-configured before rollout: half of them run the code for symmetric additive privacy homomorphism and asymmetric additive privacy homomorphism, and the remaining half of the sensor nodes run order-preserving encryption scheme, and both types of nodes are equally distributed over the covered region, which limits its practice in real-world applications. In addition, node compromise attacks are still a challenge to this method. Similarly, Secure-Enhanced Data Aggregation (SEDA) elliptic curve cryptography (ECC) [9] was developed following the principles of homomorphic encryption and the divide-and-conquer strategy. An aggregation tree disjoint method is used to divide the tree into three subtrees of similar sizes, and a homomorphic encryption-based aggregation is performed in each subtree to generate an aggregated partial result. The base station verifies the aggregated results of subtrees by comparing the count of ciphertexts.

To gain greater security, longer keys are usually used, which contradicts to the aim of minimizing energy consumption. To address key size issue without degrading the security strength, ECC has been employed for generating keys in WSN applications. To overcome great computational overhead of the asymmetric cryptography, Boneh *et al*. [10] presented a method that relies on a cyclic group of elliptic curve points. Most recently, Biswas *et al*. proposed an encryption method based on ECC and chaotic map [5]. The method employs elliptic curve points to verify the sensor nodes and use as one of the chaotic map parameters to generate the pseudorandom bit sequence. This sequence is then used in computation to encrypt data.

To deal with dynamics in cluster-based WSNs, SLEACH [11] used symmetric hash chains and symmetric operations to achieve low energy consumption. Yet, the clusters remain unchanged throughout the life of the network. Because of the symmetric mechanism, a compromised node could devastate the entire network. Oliveira *et al*. developed SecLEACH [12] that integrates random key pre-distribution and authenticated broadcast to secure hierarchical WSNs with dynamic cluster formation. In the random key pre-distribution, each node is assigned a set of keys drawn from a large key pool. When a node takes a role of cluster head, it includes information of the keys in its key ring. The authentication is performed by the base station. Lu *et al*. proposed SET-IBS and SET-IBOOS protocols [13] for clustered WSN, which leverage the identity-based digital signature scheme and the identity-based online/offline digital signature schemes. In SET-IBS, security relies on the hardness of the Diffie–Hellman problem in the pairing domain, whereas SET-IBOOS relies on the hardness of the discrete logarithm problem. Among these methods, key generation is usually computational expensive, which limits the strength of the key and the dynamics of the network structure.

Ensuring security requires extra energy in addition to the amount for sustaining the fundamental data acquisition and transmission functions of a WSN. How to minimize energy cost for encrypting messages is still an open challenge. This becomes a more critical issue for the cluster-based WSN because data are aggregated by the cluster head and, hence, a greater amount of energy is usually consumed for the process that leads to a shorter life of the sensor node. Hence, balancing the network security and energy consumption among all nodes must be further investigated.

In this paper, we propose an encryption scheme to secure data transmission in WSN with dynamically clustered sensor nodes. Our method extends GASONeC [14] and employs ECC to generate binary strings, which is combined with network properties to form strong encryption keys yet favorably demand low energy consumption. The key consists of four independent components of various lengths that form a total of 176 bits. The most computational expensive part of generating a key is using ECC to create a random 128-bit string. The encryption and decryption methods are built upon three operations: exclusive OR, substitution, and permutation, which also facilitate efficient computation.

The contribution of this work is twofold. First, a novel encryption key generation method is proposed that leverages ECC and include the random numbers generated dynamically from the network execution status. The modest computation to obtain the encryption key demands low energy consumption that enables longer network life. The proposed method randomly selects secret keys for every session rather than generating ones based on fixed parameters, which makes the security system harder to break by adversaries. Second, data aggregation at the cluster heads requires no decryption process. The integration of homomorphic encryption not only avoids attacks by the compromised cluster head but also saves energy by eliminating the recovery of the data from the cluster member nodes.

The rest of this article is organized as follows: Section 2 describes our proposed method to secure dynamically clustered sensors in WSN. The section starts with a description of encryption key generation followed by detailed explanation of data encryption/decryption process. Section 3 first analyzes the security aspects of our methods and then presents the experimental results in comparison with three state-of-the-art methods. Section 4 concludes this article with a summary and gives future work.

2. METHODOLOGY

Figure 1 illustrates a taxonomy of encryption methods, and a couple of algorithms are given for each sub-category. Among the three categories of cryptographic methods, symmetric key ciphers can be designed to have high rates of data throughput, and the key is relatively short. However, the sheer volume of key pairs for two-party communication is a challenge faced in the application to WSN. On the other hand, the throughput rate for the public key-based methods is several orders of magnitude

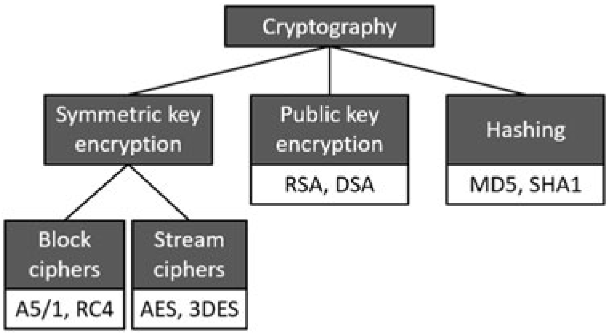


Figure 1. Taxonomy of encryption methods.

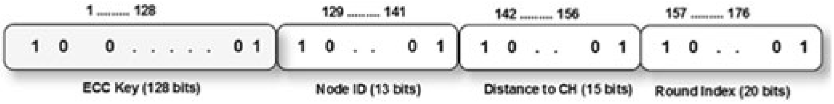


Figure 2. Our proposed encryption key consists of four components: an ECC key, node ID, distance to cluster head (CH), and the round index.

slower, and the key sizes are typically much larger. A hash function is a computationally efficient function mapping binary strings of arbitrary length to binary strings of a fixed length. Hence, encryption methods using hash function as primitive are usually computationally favorable in resource-limited applications [15].

In the conventional cryptographic methods, key generation and exchange require great energy consumption and communication overhead. To minimize such overhead, ECC is used to generate concise keys for each sensor node. This key and the sensor node properties are used to encrypt data without altering its size. Our proposed encryption uses random bit substitution and regulated permutation to generate ciphertexts, which are aggregated by the cluster head and forwarded to the base station. The base station disassembles and decrypts the data using the agreed keys retrieved via a hashing function. In each transmission round, a unique encryption key is selected for each sensor to ensure data confidentiality.

Our method is built upon GASONeC [14], which employs genetic algorithm to optimize sensor clusters dynamically for balancing energy consumption and to extend network life. In each transmission round, network structure is updated according to the combined factors including expected energy expenditure, local sensor density, remaining energy, and distance to the base station. To ensure security in such a dynamically clustered sensor network with limited energy, the security protocol must take energy consumption into consideration.

2.1. Key generation and initialization

Following the ECC algorithm, a pool of binary strings is created for each sensor, which is also stored in the base station.[†] Note that the ECC parameters are preloaded to the sensors and only the elliptic curve points are saved, which require fairly small storage space. When a binary string is needed, a sensor randomly selects an elliptic curve point from this pool and maps it with a hash function. This hashed key is shared with the base station, which is used to retrieve the matching binary string from the corresponding pool. This process not only hides the explicit binary string from the network transmission but also facilitates a means of verifying the legitimacy of the source node.

Besides the binary string generated from ECC, our proposed encryption key contains three more components. The structure of a key is the following. The first component consists of 128 bits produced with the ECC algorithm. The second component represents the node ID in 13 bits. The third component consists of 15 bits to encode the distance between the sensor and its cluster head, which is followed by 20 bits to represent the count of transmission rounds. In our scheme, each sensor node is preloaded with the ECC parameters and is initialized with an ID. The structure of a key is shown in Figure 2.

During the network initialization phase, because sensor node and the base station know the transmission round, only the binary string from ECC and the ID are hashed and transmitted from a sensor to the base station. The acknowledgment from the base station returns a confirmation message and the distance from the sensor to its cluster head. The hashed code of the keys is produced using SHA-2 hashing function.

[†] Our encryption key contains more than what is generated from elliptic curve cryptography. To avoid confusion, we use binary string to refer to the key resulted from elliptic curve cryptography algorithm.

DOI: 10.1002/sec

Our proposed encryption key consists of 176 bits. Hence, the possible number of keys is 2^{176}, which is approximately 9.58×10^{52}. Hence, a brute-force attack is unlikely to succeed for such a large key space.

2.2. Data encryption and decryption

Data are encrypted at the sensor nodes and decrypted only at the base station. Cluster head aggregates data from its member nodes without decryption. Our encryption and decryption method employs three simple operations to enforce confusion and diffusion: exclusive OR operation, substitution, and permutation. To facilitate these operations, the encryption key is divided into two equal parts. The input data are of the same length as half of the encryption key in terms of the number of bits.

2.2.1. Exclusive OR.

In this operation, the first part of the encryption key, namely, the randomize bits, and the input data are the operands of exclusive OR.

2.2.2. Substitution.

In this operation, the second part of the encryption key, namely, the reference bits, is used to guide the substitution operation in the output from the exclusive OR operation. The count of ones in every 8 bit of the reference bits is used to decide the starting point of the two consecutive bits in a byte of the data to alter their values. For instance, assume that the counts of 8 bit in the reference bits are {2 4 3 3 3 0 1 1 3 0 3} and the output from the exclusive OR operation is

$$\{0000011100001101000011000100000001010111\ldots\}.$$

For illustration purpose, only the first 40 bits are shown. The results of substitution are

$$\{0\underline{11}00111000\underline{10}10100\underline{11}110001\underline{11}000001\underline{10}0111\ldots\}.$$

The underlined bits are altered in the substitution operation.

2.2.3. Permutation.

In this operation, again, the second part of the encryption key is used to guide the permutation operation in the output from the substitution operation. The count of ones in every 11 bit of the reference bits decides which two bytes in a 4-byte segment are switched. If two consecutive counts are in the increasing order, the third and fourth bytes are switched; otherwise, the first and second bytes are switched. Assume that the counts of 11 bit in the reference bits are {4 5 3 3 1 3 1 3}. Using the output from the previous substitution example, the first iteration takes 4 bytes

$$\{01100111\ 00010101\ 00111100\ 01110000\}.$$

Because the first two 11-bit counts in the reference bits are {4 5}, which is in ascending order, the third and fourth bytes are switched, which yields

$$\{01100111\ 00010101\ \underline{01110000}\ \underline{00111100}\}.$$

In the next iteration, the four consecutive bytes are

$$\{00010101\ \underline{01110000}\ \underline{00111100}\ 01100111\}.$$

Note that the underlined bytes are switched in the previous iteration. Because the reference counts {5 3} are in

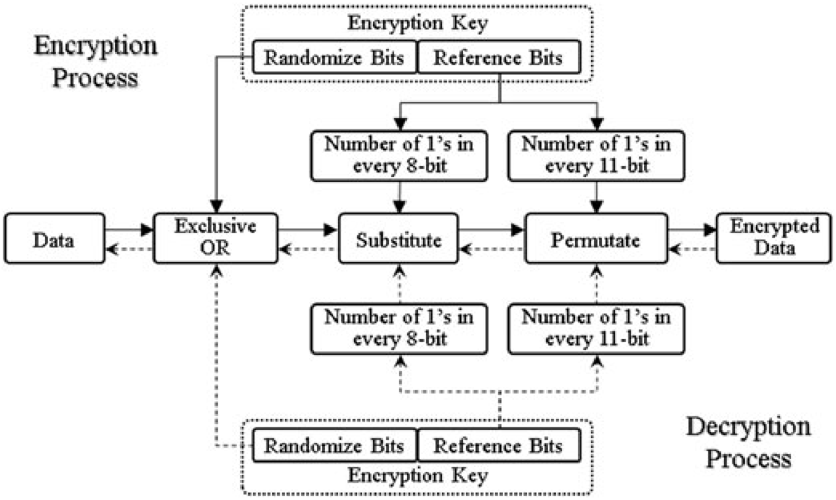


Figure 3. Encryption and decryption process. The solid arrows show the data flow in the encryption process, and the dash arrows show the data flow in the decryption process. OR,.

DOI: 10.1002/sec

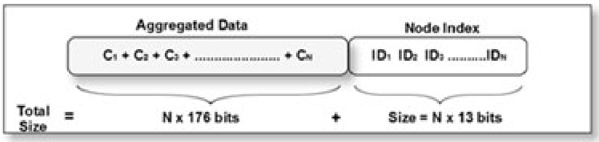


Figure 4. An illustration of the aggregated data at the cluster head. *N* is the number of nodes in the cluster. The package consists of the encrypted data and the node IDs.

descending order, the first two bytes are switched, which yields

$$\{\underline{01110000}\ \underline{00010101}\ 00111100\ 01100111\}.$$

This process repeats until the end of input data.

Figure 3 illustrates the encryption and decryption process of our proposed method. The decryption process follows the inverse order of the three operations, and the data flow is depicted with dash arrows. Note that the permutation operation in the decryption process uses the 11-bit counts in the inverse order and processes the encrypted data from the end to the front. The arrows show the process flow: The solid arrows present the encryption procedure, and the dash arrows present the decryption procedure.

2.3. Data aggregation and recovery

Data aggregation at a cluster head is straightforward. The encrypted data from member nodes are concatenated by the cluster head together with the node IDs as shown in Figure 4. The fixed size of each unit allows ease of disassembling the data package at the base station.

Following the GASONeC algorithm, the base station assigns clusters dynamically throughout the life of the WSN. With the knowledge of the cluster size, denoted with N, the base station divided the last $N \times 13$ bits of the received data from a cluster head into N pieces of 13-bit segment, each of which gives a sensor ID. Following the order of the sensor IDs, the first $N \times 176$ bits are divided into N pieces. The decryption of each piece is achieved following the algorithm described in Section 2.2.

3. SECURITY ANALYSIS AND EXPERIMENTAL RESULTS

3.1. Experimental settings

Our proposed method is simulated based on the properties of MICAz [16] sensor. The MICAz uses Atmel ATmega128L [17] that is a low-power microcontroller with a clock frequency of 7.37 MHz and runs with TinyOS. It has 128K bytes of program flash memory and embeds an IEEE 802.15.4 compliant Radio Frequency (RF) transceiver CC2420 with a data rate at 250 kbps.

The WSN consists of 100 nodes with equal initial energy of 0.5J. The nodes are randomly placed in a field of 100 m×100 m, and the base station is positioned at the field boundary. The idle state energy is 50 nJ/bit, and the energy for aggregating data is 5 nJ/bit. The amplification energy is 10 pJ/bit/m^2 when the distance is greater than d_0; otherwise, it is 0.0013 pJ/bit/M^2. The average performance of 10 repetitions is reported in our results.

The following properties apply to the WSN in all test cases:

- There is one base station that receives data from nodes.
- Nodes are stationary, and their locations are known.
- Provided with sufficient energy, each node can directly reach the base station.
- The characteristics and initial energy of each node are the same.

Because our method extends GASONeC [14] that employs genetic algorithm for optimization, we set the population size of each generation at 30 for a total of 30 generations. The crossover probability and mutation probability are 0.8 and 0.006, respectively. The neighborhood distance threshold is 20 m.

3.2. Security analysis

3.2.1. Brute-force attack.

As demonstrated in Section 2.1, our proposed encryption key consists of 176 bits, which spans a key space with a size of 9.58×10^{52}. Hence, it is infeasible for adversaries to decrypt the eavesdropped message without the key with a brute-force attack.

3.2.2. HELLO flood attack.

In this attack, adversary sends the HELLO packet and aims to consume the network resources. In our method, the sensor nodes receive and transmit data to the dynamically assigned cluster head and the network topology, and data flow is maintained by the base station. Each node is authenticated by the base station to transmit data. Hence, our method is immune to Hello flood attack. Similarly, it is unaffected by the *selective forwarding attack* because data from a sensor node are forwarded by a designated cluster head.

3.2.3. Denial of service attack.

In this attack, an adversary sends packets and utilizes network bandwidth to prevent the user from accessing the service or resources. This is overcome by changing the cluster head dynamically after each transmission round and informing the cluster head of its member nodes. In addition, an acknowledge message sent by the base station helps avoid such attacks.

3.2.4. Compromised cluster head attack.

Compromised cluster head attack is an attack that tries to aggregate the data from all cluster nodes by deluding them that it is working as a cluster head. The aim of this attack are to analyze data and to conclude specific information after receiving it. In our proposed schema, the role of cluster head is to forward the encrypted data from its cluster nodes and the base station without decrypting it. In addition, our encryption process depends on many factors, that is, node location, round index, and the distance between the node and its cluster head; this combination requires any attacker to have complete knowledge about the network operations (round index), network topology (distance between node and cluster head), and the secret binary string that is produced by ECC.

3.3. Energy consumption and network lifetime

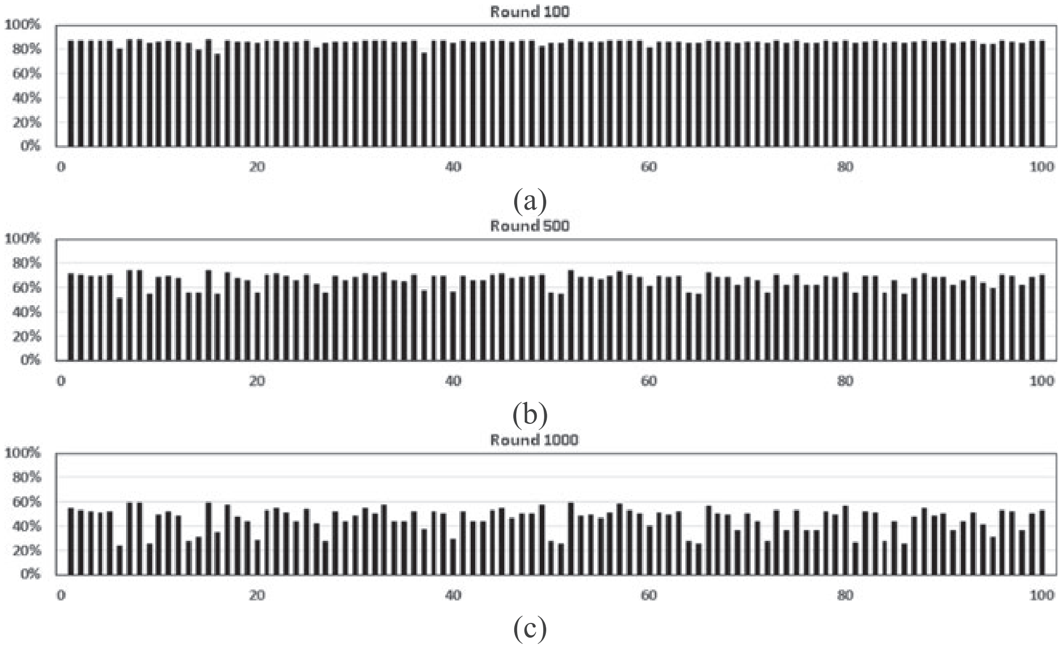
In a balanced WSN, the remaining energy in each node is of a similar amount. That is, no node exhausts its energy before the others by sharing the duties that require extra energy consumption. Figure 5 illustrates the percentage of the remaining energy of sensor nodes at transmission rounds of 100, 500, and 1000 using our proposed method. The *x*-axis shows the index of nodes, and the *y*-axis shows the percentage of the remaining energy at each node. As described in Section 3.1, the initial amount of energy of all nodes is same. At the 100 iteration, the remaining energy of all nodes is almost identical. As the network transmission proceeds, the remaining energy of all nodes reduces, but the difference is relatively small. At the 1000 iteration, some of the nodes deplete energy at a higher rate, yet the energy of the majority is in a close range. The fluctuation is partly attributed to the sub-optimal solution identified by the genetic algorithms when limited number of evolution iterations is performed to satisfy real-time need. It is evident that the remaining energy of all sensors remains very close throughout the life of the WSN and there are some fluctuations towards the end of the network life.

Table I. The average and standard deviation of the remaining energy for six simulation runs.

Rounds	100	300	500	700	900	1000
Mean	0.428	0.387	0.331	0.298	0.248	0.228
STD	0.010	0.021	0.029	0.036	0.046	0.050

SD, standard deviation.

The same trend of the remaining energy among sensor nodes can also be observed in Table I, which presents the mean and standard deviation of the remaining energy of sensor nodes in a network at ten different simulations. In each simulation case, the same settings were used for the WSN, and the remaining energy was documented at each iteration. It is shown that the remaining energy of all sensor nodes decreases mostly evenly with a small variation.



(a)

(b)

(c)

Figure 5. The remaining energy of all sensor nodes at network transmission round 100 (a), 500 (b), and 1000 (c).

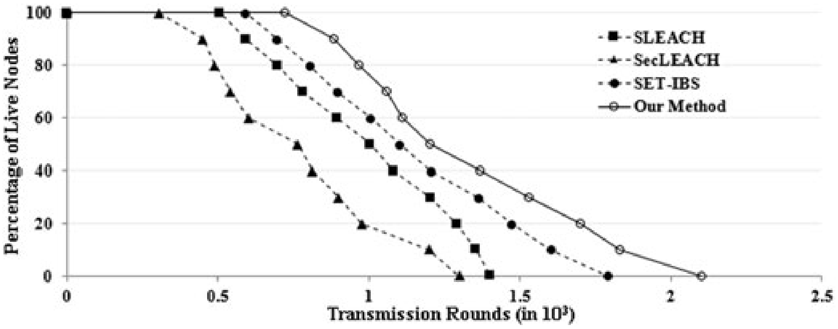


Figure 6. Network lifetime shows the percentage of the remaining number of live sensor nodes given the base station is placed at the corner of the field. The *x*-axis shows the number of transmission rounds in the order of 10^3.

Table II. The number of transmission rounds when the respective percentage of sensor nodes remains available in the network.

Live node	SecLEACH	SLEACH	SET-IBS	Our Method
100%	306	509	591	723
90%	450	592	695	885
80%	490	698	801	965
70%	542	780	894	1060
60%	601	893	1005	1110
50%	764	1004	1100	1202
40%	814	1082	1203	1370
30%	901	1204	1360	1530
20%	976	1290	1473	1707
10%	1200	1354	1600	1830
0%	1300	1402	1790	2103

Table III. The number of transmission rounds that took 10% of all nodes in a wireless sensor network to exhaust their energy and become unavailable using different encryption and network clustering methods.

Percentage	SecLEACH	SLEACH	SET-IBS	Our method
100 to 90	144	83	104	**162**
90 to 80	40	**106**	**106**	80
80 to 70	52	82	93	**95**
70 to 60	59	**113**	111	50
60 to 50	**163**	111	95	92
50 to 40	50	78	103	**168**
40 to 30	87	122	157	**160**
30 to 20	75	86	113	**177**
20 to 10	**224**	64	127	123
10 to 0	100	48	190	**273**
Average	99.4	89.3	119.9	**138**

The variation, represented as standard deviation, increased as the transmission continues, but remains fairly low.

In our comparison study, we adopt three state-of-the-art methods, SLEACH [11], SecLEACH [12], and SET-IBS [13]. The average network life in terms of percentage of live nodes for each method is illustrated in Figure 6. It is clear that our proposed method consistently yielded the greatest average number of transmission rounds throughout the life of the WSN. This indicates that our proposed security method demands less amount of energy for key generation and network maintenance overhead. Among the three comparative methods, SET-IBS resulted in the most competitive network life.

The readings of the live nodes are presented in Table II. In contrast to the method with second best performance (i.e., SET-IBS), the improvement of network life is 22.3% when all nodes are alive, and is 17.5% when the last node deceased. It is evident that our proposed method greatly extends the network life of a secure WSN.

It is also interesting to analyze the gracefulness of the WSN life. Table III presents the number of transmission rounds that took 10% of all nodes in a WSN to exhaust their energy and become unavailable using different encryption and network clustering methods. The largest numbers of transmission rounds are highlighted with bold font. Our method exhibited the largest average number of transmission rounds to exhaust the energy of sensor nodes, which implies the most graceful degradation of the network function. SET-IBS ranked the second on average, which is consistent to its performance in terms of overall network life. It is noteworthy that despite that SLEACH sustained a longer network life than SecLEACH, it took SLEACH less number of transmission rounds for the sensors to discontinue on average.

4. CONCLUSION

The dynamic nature of sensor network and numerous possible cluster configurations make searching for a secure and optimal network structure an open challenge. In this paper, we propose a novel encryption method to secure data transmission in WSN with dynamic sensor clusters. Our method leverages ECC algorithm to generate binary strings for each sensor and combines with node ID, distance to its cluster head, and the round index to form unique 176-bit encryption keys. Using exclusive OR, substitution, and permutation operations, encryption is achieved efficiently.

Compared with the state-of-the-art methods, our results demonstrated that the proposed method exhibits much improved network lifetime and reduces the energy consumption most evenly among all sensor nodes. The numerical example illustrates the security strength of our encryption method. More importantly, our method overcomes a number of attacks including brute-force attack, HELLO flood attack, selective forwarding attack, and compromised cluster head attack, which are discussed via security analysis.

In our future work, we plan to extend our method and explore the decentralized means of creating network structure and data encryption. For a WSN that covers a large area, communicating directly to the base station is energy demanding. A decentralized means of organizing sensor nodes for data acquisition, encryption, and transmission could benefit the robustness of the network and shield it from possible breach at a few nodes. In addition, the multi-hop communication will be investigated, and its influence to the network life will be studied.

REFERENCES

1. Li B, Li H, Wang W, Yin Q, Liu H. Performance analysis and optimization for energy-efficient cooperative transmission in random wireless sensor network. *IEEE Transactions On Wireless Communications* 2013; **12**(9): 4647–4657.
2. Mykletun E, Girao J, Westhoff D. Public key based cryptoschemes for data concealment in wireless sensor networks, *The IEEE International Conference On Communications*, IEEE, Istanbul, 2006; 2288–2295.
3. Liu Y, Tian S. Design and statistical analysis of a new chaos block cipher for WSN. *Communications in Nonlinear Science and Numerical Simulation* 2012; **17**(8): 3267–3278.
4. Isobe T, Shibutani K. Security analysis of the lightweight block ciphers XTEA and LED and Piccolo. In *Information Security and Privacy*. W. Susilo, Y. Mu, and J. Seberry (Eds.): ACISP 2012, LNCS 7372, pp. 71–86, 2012.
5. Biswas K, Muthukkumarasamy V, Singh K. An encryption scheme using chaotic map and genetic operations for wireless sensor networks. *IEEE Sensors Journal* 2015; **15**(5): 2801–2809.
6. Suzaki T, Minematsu K, Morioka S, Kobayashi E. TWINE: a lightweight block cipher for multiple platforms. *Selected Areas in Cryptography* 2013; **7707**: 339–354.
7. Karakoca F, Demircia H, Harmancib A. Biclique cryptanalysis of LBlock and TWINE. *Information Processing Letters* 2013; **113**(12): 423–429.
8. Girao J, Westhoff D, Mykletun E, Araki T. TinyPEDS tiny persistent encrypted data storage in asynchronous wireless sensor networks. *Ad Hoc Networks* 2007; **5**(7): 1073–1089.
9. Zhou Q, Yang G, He L. A secure enhanced data aggregation based on ECC in wireless sensor network. *Sensors Journal* 2014; **14**(4): 6701–6721.
10. Boneh D, Goh E, Nissim K. Evaluating 2-DNF formulas on ciphertexts, *Proceeding of the 2nd international conference on theory of cryptography*, 2005; 325–341.
11. Xiao W, Li Y, Ke C. SLEACH secure lowenergy adaptive clustering hierarchy protocol for wireless sensor networks. *Wuhan University Journal of Natural Sciences* 2005; **10**(1): 127–131.
12. Oliveira L, Ferreira A, Vilaca M, Wong H, Bern M, Dahab R, Loureiro A. SecLEACH - on the security of clustered sensor networks. *Signal Processing* 2007; **87**(12): 2882–2895.
13. Huang L, Jie L, Guizani M. Secure and efficient data transmission for cluster-based wireless sensor networks. *IEEE Transactions on Parallel and Distributed Systems* 2014; **25**(3): 750–761.
14. Elhoseny M, Yuan X, Yu Z, Mao C, El-Minir H, Riad A. Balancing energy consumption in heterogeneous wireless sensor networks using genetic algorithm. *IEEE Communications Letters* in press.
15. Elhoseny M, El-Minir HK, Riad AM, Yuan X. Balancing energy consumption in heterogeneous wireless sensor networks using genetic algorithm. *International Journal of Computer Networks and Communications Security* 2014; **2**(11): 400–413.
16. MEMSIC. http://www.memsic.com/userfiles/files/Datasheets/WSN/micaz_datasheet-t.pdf, accessed in November 2015.
17. ATMEL. http://www.atmel.com/images/2467s.pdf, accessed in November 2015.